Claim Amendments

Claim 1 (currently amended): An apparatus for data storage comprising:

a cluster of NFS (network file system) servers, each server having network ports for incoming file system requests and cluster traffic between servers, each server has a network element and a disk element; and

a plurality of storage arrays in communication with the servers, the servers utilizing a striped file system for storing data for providing bandwidth to multiple disk elements, where the striped file system comprises a set of striped VFSes (virtual file systems) distributed among a number of disk elements of the cluster of servers, with one VFS of the set of striped VFSes per disk element, wherein a data file is striped among all the VFSes of the set of striped VFSes with different strips of the file's data in different VFSes in the set of striped VFSes.

Claim 2 (canceled)

Claim 3 (previously presented): An apparatus as described in Claim 1 wherein each disk element has a virtual file system with the virtual file system of each disk element together forming a striped VFS.

Claim 4 (currently amended): An apparatus as described in Claim [[3]] 45 wherein all disk elements for a virtual file system act as meta-data servers.

Claim 5 (original): An apparatus as described in Claim 4 wherein a file has attributes and each server for each file maintains a caching element that stores a last known version of the file attributes and ranges of modification time and change time values for assignment to write operation results.

Claim 6 (original): An apparatus as described in Claim 5 wherein each disk element which is not the meta-data server for a virtual file system is an input output secondary.

Claim 7 (original): An apparatus as described in Claim 6 wherein ranges of file modification times or file change times are reserved from the meta-data server by the input output secondary.

Claim 8 (original): An apparatus as described in Claim 7 wherein the modification and change times in the ranges obtained from the meta-data server are issued to operations already queued at the input output secondary.

Claim 9 (original): An apparatus as described in Claim 8 wherein modification and change times in the ranges obtained from the meta-data server are issued to operations received during a window of time after the ranges are reserved from the meta-data server by the input output secondary.

Claim 10 (original): An apparatus as described in Claim 9 wherein operations affecting all stripes of a file begin executions first at the meta-data server for a file, and then execute at all input output secondaries, such that operations at the input output secondaries wait only for already executing operations that have already finished their communication with the meta-data server.

Claim 11 (original): An apparatus as described in Claim 10 wherein operations follow one of at least two locking models, the first of which is to synchronize first with the meta-data server, then begin core execution by synchronizing with other operations executing at the input output secondary, and the second of which is to first synchronize at the meta-data

server, and then to synchronize with operations at one or more input output secondaries that have begun core execution at the input output secondaries.

Claim 12 (original): An apparatus as described in Claim 11 wherein the cluster network is connected in a star topology.

Claim 13 (original): An apparatus as described in Claim 12 wherein the cluster network is a switched Ethernet.

Claim 14 (currently amended): A method for data storage comprising the steps of:

creating a file across a plurality of NFS (Network File System) servers, each server having a network element and a disk element;

writing data into the file as strips of the data in the servers, the strips together forming a stripe for providing bandwidth to multiple disk elements, where the striped file system comprises a set of striped VFSes (virtual file systems) distributed among a number of disk elements of the cluster of servers, with one VFS of the set of striped VFSes per disk element;

reading the strips of the data from the servers; and

deleting the strips from the servers.

Claim 15 (currently amended): A method as described in Claim [[14]] 47 including the step of identifying a disk element for a virtual file system of an NFS [[a]] server as a meta-data server and disk elements for the NFS servers ~~severs~~ which are not identified as the meta-data server as input output secondaries.

Claim 16 (original): A method as described in Claim 15 including the step of storing in a caching element at each input output secondary for each active file at a meta-data server a last known version of attributes of the file which are good for a dallying period.

Claim 17 (original): A method as described in Claim 16 including the step of storing ranges of modification time and change time values in the caching element for assignment to write operations.

Claim 18 (original): A method as described in Claim 17 including the step of making a status request by the caching element to the meta-data server to obtain a file's current attributes.

Claim 19 (original): A method as described in Claim 18 wherein the making a status request step includes the step of obtaining modification time and change time ranges from the meta-data server.

Claim 20 (original): A method as described in Claim 19 including the step of queuing file read and file write requests at the input output secondary until the file read and file write requests are admitted by the cache element and complete execution.

Claim 21 (original): A method as described in Claim 20 including the step of tracking by the cache element of the file read and file write requests executing for the file and the ranges that are being read or written.

Claim 22 (original): A method as described in Claim 21 including the step of requesting the cache element move out of invalid node to read mode when a read operation must be executed.

Claim 23 (original): A method as described in Claim 22 including the step of checking a byte range affected by a file read request to ensure it does not overlap a byte range of any file write requests previously admitted and currently executing.

Claim 24 (original): A method as described in Claim 23 including the step of requesting, in response to a file write request that the cache element move into a write mode.

Claim 25 (original): A method as described in Claim 24 including the step of checking with the cache element the byte range affected by the file write request for overlap with any admitted and still executing file read or file write requests.

Claim 26 (original): A method as described in Claim 25 including the step, when executing a write request, of allocating a modification time and change time pair from the range of modification times and change times stored in the cache element.

Claim 27 (original): A method as described in Claim 26 including the step of checking the head of a queue of pending file read and file write requests to see if a head request can be admitted by the caching element after either a file read or file write request is completed.

Claim 28 (original): A method as described in Claim 27 including the steps of detecting by the cache element that a file length must be updated in response to a file write request, moving the cache element into exclusive mode; and making a file write status call to the meta-data server to update length attributes of the file.

Claim 29 (original): A method as described in Claim 14 including the step of storing in a caching element at each input output secondary for each active file at a meta-data server a last known version of attributes of the file which are good for a dallying period.

Claim 30 (currently amended): A method as described in Claim [[14]] 47 including the step of storing ranges of modification time and change time values in a caching element for assignment to write operations.

Claim 31 (currently amended): A method as described in Claim [[14]] 47 including the step of making a status request by a caching element to the meta-data server to obtain a file's current attributes.

Claim 32 (original): A method as described in Claim 31 wherein the making a status request step includes the step of obtaining modification time and change time ranges from the meta-data server.

Claim 33 (currently amended): A method as described in Claim [[14]] 47 including the step of requesting a cache element move out of invalid node to read mode when a read operation must be executed.

Claim 34 (currently amended): A method as described in Claim [[14]] 47 including the step of requesting, in response to a file write request that a cache element move into a write mode.

Claim 35 (currently amended): A method as described in Claim [[14]] 47 including the steps of detecting by a cache element that a file length must be updated in response to a file write request, moving the cache element into exclusive mode; and making a file write status call to a meta-data server to update length attributes of the file.

Claim 36 (currently amended): A method for establishing storage for a file comprising the steps of:

receiving an NFS (network file system) create request at a network element;

receiving a file create request at a meta-data server from the network element;

allocating an inode number for the file at the meta-data server;

making create calls to input output secondaries to mark the file as allocated by the input output secondaries; and

committing the file create at the meta-data server.

Claim 37 (original): A method for removing a file from storage comprising the steps of:

receiving a delete file request at a meta-data server;

removing a file name of the file from a parent directory by the meta-data server at the meta-data server;

putting the file on a file delete list by the meta-data server at the meta-data server;

sending delete calls to the input output secondaries;

receiving at the meta-data server acknowledgment calls from the input output secondaries that they have deleted the file;

deleting the file at the meta-data server;

removing the file from the file delete list; and

placing an inode number associated with the file into a free list by the meta-data server at the meta-data server.

Claim 38 (currently amended): A method for reading data in a file comprising the steps of:

receiving an NFS (network file system) read request for data in the file at a network element;

determining by the network element which VFS stores at least one strip containing the data;

sending a file read request from the network element to at least one disk element of a plurality of servers storing a strip of the data;

obtaining current attributes associated with the file by each disk element;

reading the strips of the file from each disk element having the strips; and

generating a response in regard to the file read request.

Claim 39 (currently amended): A method for writing data in a file comprising the steps of:

receiving an NFS (network file system) write request for a file at a network element;

determining by the network element which VFS (virtual file system) is associated with the file;

sending a file write request from the network element to at least one disk element of a plurality of servers having a stripe of the VFS;

acquiring current attributes associated with the file; and

writing a predetermined number of bytes of the data into each VFS strip in succession until all of the data is written into the file.

Claim 40 (canceled)

Claim 41 (currently amended): A method as described in Claim [[14]] 47 including the step of identifying a disk element for a virtual file system of an NFS (network file system) server as a meta-data server and disk elements for the NFS severs which are not identified as the meta-data server as input output secondaries.

Claim 42 (new): An apparatus as described in Claim 3 wherein the different VFSes in the set of striped VFSes have a same vnode number.

Claim 43 (new): An apparatus as described in Claim 42 wherein a strip N of vnode B in the set of striped VFSes is stored on an I-th server where I = (B+N) and STRIPE_WIDTH and STRIPE_WIDTH is a number of strips in a striped VFS across all storage arrays holding the striped VFS.

Claim 44 (new): An apparatus as described in Claim 43 wherein one server of the cluster of servers is a meta-data server for one of the striped VFSes.

Claim 45 (new): A method as described in Claim 44 wherein all files of the one of the striped VFSes are represented at the meta-data server.

Claim 46 (new): A method as described in Claim 14 wherein the writing step includes the step of writing the strips in different VFSes in the set of striped VFSes having a same vnode number.

Claim 47 (new): A method as described in Claim 46 wherein the writing step includes the step of storing a strip N of vnode B in the set of striped VFSes on an I-th server where I = (B+N) and STRIPE_WIDTH and STRIPE_WIDTH is a number of strips in a striped VFS across all storage arrays holding the striped VFS.

Claim 48 (new): An apparatus for data storage comprising:

a cluster of network-accessed file level servers, each server having network ports for incoming file system requests and cluster traffic between servers, each server has a network element and a disk element; and

a plurality of storage arrays in communication with the servers, the servers utilizing a striped file system for storing data, and where one disk element for a given file system acts as a meta-data server that maintains modification and change time attributes for each file, and where each server for each file maintains a caching element that stores a last

known version of the file attributes and ranges of modification time and change time values for assignment to write operation results.

Claim 49 (new): An apparatus as described in Claim 48 wherein each disk element which is not the meta-data server is an input output secondary.

Claim 50 (new): An apparatus as described in Claim 49 wherein ranges of file modification times or file change times are reserved from the meta-data server by the input output secondary.

Claim 51 (new): An apparatus as described in Claim 50 wherein the modification and change times in the ranges obtained from the meta-data server are issued to operations already queued at the input output secondary.

Claim 52 (new): An apparatus as described in Claim 51 wherein modification and change times in the ranges obtained from the meta-data server are issued to operations received during a window of time after the ranges are reserved from the meta-data server by the input output secondary.

Claim 53 (new): An apparatus as described in Claim 52 wherein operations affecting all stripes of a file begin executions first at the meta-data server for a file, and then execute at all input output secondaries, such that operations at the input output secondaries wait only for already executing operations that have already finished their communication with the meta-data server.

Claim 54 (new): An apparatus as described in Claim 53 wherein operations follow one of at least two locking models, the first of which is to synchronize first with the meta-data server, then begin core execution by synchronizing with other operations executing at the input output secondary, and the second of which is to first synchronize at the meta-data server, and then to synchronize with operations at one or more input output secondaries that have begun core execution at the input output secondaries.

Claim 55 (new): An apparatus as described in Claim 54 wherein the cluster network is connected in a star topology.

Claim 56 (new): An apparatus as described in Claim 55 wherein the cluster network is a switched Ethernet.

Claim 57 (new): An apparatus as described in Claim 56 wherein the servers are NFS servers.

Claim 58 (new): A method for data storage comprising the steps of:

receiving incoming file system requests at network ports of a cluster of network-accessed file level servers, and cluster traffic between servers at the ports, each server has a network element and a disk element; and

storing data utilizing a striped file system in a plurality of storage arrays in communication with the servers, where one disk element for a given file system acts as a meta-data server that maintains modification and change time attributes for each file, and where each server for each file maintains a caching element that stores a last known version of the file attributes and ranges of modification time and change time values for assignment to write operation results.

Claim 59 (new): A method as described in Claim 58 wherein each disk element which is not the meta-data server is an input output secondary, and wherein the storing step includes the step of reserving ranges of file modification times or file change times from the meta-data server by the input output secondary.

Claim 60 (new): A method as described in Claim 59 wherein the storing step includes the step of issuing the modification and change times in the ranges obtained from the meta-data server to operations already queued at the input output secondary.

Claim 61 (new): A method as described in Claim 60 wherein the issuing step includes issuing the modification and change times in the ranges obtained from the meta-data server to operations received during a window of time after the ranges are reserved from the meta-data server by the input output secondary.

Claim 62 (new): A method as described in Claim 61 including the step of executing operations affecting all stripes of a file beginning first at the meta-data server for a file, and then executing at all input output secondaries, such that operations at the input output secondaries wait only for already executing operations that have already finished their communication with the meta-data server.

Claim 63 (new): A method as described in Claim 62 including the step of executing operations following one of at least two locking models, the first of which is to synchronize first with the meta-data server, then begin core execution by synchronizing with other operations executing at the input output secondary, and the second of which is to first

synchronize at the meta-data server, and then to synchronize with operations at one or more input output secondaries that have begun core execution at the input output secondaries.

Claim 64 (new): A method as described in Claim 63 wherein the servers are NFS servers.